Centre de recherche sur les transports (C.R.T.)
Centre for Research on Transportation

CRT-2005-03

THE BLACK AND WHITE TRAVELING SALESMAN PROBLEM

by

Gianpaolo Ghiani
Gilbert Laporte
Frédéric Semet

February 2005

Université de Montréal
C.P. 6128, succursale Centre-ville, Montréal QC H3C 3J7 Canada
Téléphone: (514) 343-7575 / Télécopieur: (514) 343-7121

crt@crt.umontreal.ca
www.crt.umontreal.ca

HEC ÉCOLE POLYTECHNIQUE MONTRÉAL Université de Montréal

105-03305/

PUBLICATION
CRT-2005-03

THE BLACK AND WHITE TRAVELING SALESMAN PROBLEM

by

Gianpaolo Ghiani[1]
Gilbert Laporte[2]
Frédéric Semet[3]

The research was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC).

* Revised version of CRT-99-47.

1. Dipartimento di Ingegneria dell'Innovazione, Università di Lecce, Via per Arnesano, 73100 Lecce, Italy

2. Centre de recherche sur les transports, Université de Montréal, C.P. 6128, succursale Centre-ville, Montréal, Canada H3C 3J7 and Canada Research Chair in Distribution Management, HEC Montréal, 3000 Côte-Sainte-Catherine, Montréal, Canada H3T 2A7

3. LAMIH, Université de Valenciennes et du Hainaut-Cambrésis, Le Mont-Houy, 59311 Valenciennes Cedex 9, France

Dépôt légal – Bibliothèque nationale du Québec
Bibliothèque nationale du Canada, 2005

Abstract

The *Black and White Traveling Salesman Problem* (BWTSP) is defined on a graph G whose vertex is partitioned into black and white vertices. The aim is to design a shortest Hamiltonian tour on G subject to cardinality and length constraints: both the number of white vertices as well as the length of the tour between two consecutive black vertices are bounded above. The BWTSP has applications in airline scheduling and in telecommunications. This article proposes an integer linear formulation for the undirected BWTSP, as well as several classes of valid inequalities. An exact branch-and-cut algorithm is then developed. Extensive tests show that it can solve exactly instances involving up to 100 vertices. The algorithm can also be applied directly to solve unit demand vehicle routing problem of similar sizes.

Key words: Traveling salesman problem, vehicle routing problem, valid inequalities, branch-and-cut, exact algorithm.

Résumé

Le *problème du voyageur de commerce avec sommets blancs et noirs* (PVCBN) est défini sur un graphe G dont l'ensemble des sommets est partitionné en sommets noirs et blancs. On doit déterminer un tour Hamiltonien sur G sous deux contraintes : le nombre de sommets blancs ainsi que la longueur du tour entre deux sommets noirs consécutifs sont tous deux bornés supérieurement. On retrouve des applications du PCVBN dans la fabrication d'horaires d'avions et en télécommunications. Cet article propose une formulation par programmation linéaire en nombres entiers pour le PVCBN ainsi que plusieurs classes d'inégalités valides. On développe ensuite un algorithme de séparation et coupes. Des tests numériques extensifs indiquent qu'on peut résoudre exactement des instances du PVCBN comportant jusqu'à 100 sommets. L'algorithme s'applique aussi directement au problème de tournées de véhicules avec demandes unitaires et peut résoudre des instances de tailles similaires.

Mots clés : problème du voyageur de commerce, problème de tournées de véhicules, inégalités valides, séparation et coupes, algorithme exact.

The *Black and White Traveling Salesman Problem* (BWTSP) is defined on a directed graph $G = (V, A)$ or on an undirected graph $G = (V, E)$, where $V = \{v_1, ..., v_n\} = B \cup W$ is a vertex set partitioned into a set B of *black vertices* ($|B| \geq 2$) and a set W of *white vertices*, $A = \{(v_i, v_j) : v_i, v_j \in V, i \neq j\}$ is an arc set, and $E = \{(v_i, v_j) : v_i, v_j \in V, i \neq j\}$ is an edge set. In the undirected case the two notations (v_i, v_j) and (v_j, v_i) represent the same edge. A distance matrix (c_{ij}) is defined on A or E. The BWTSP consists of determining a shortest Hamiltonian circuit or cycle on G subject to i) *cardinality constraints:* the number of white vertices on "black-to-black" paths or chains (linking two consecutive black vertices) is bounded above by a positive integer Q, and ii) *length constraints:* the length of any path or chain between two consecutive black vertices is bounded above by a positive value L. The BWTSP reduces to the *Traveling Salesman Problem* (TSP) whenever $Q = L = \infty$ and is therefore NP-hard. Another particular case of the BWTSP is the *Vehicle Routing Problem* (VRP) with unit demands, vehicle capacity Q and maximal route length L. Using a transformation similar to that of Lenstra and Rinnooy Kan (1975) this case is obtained by locating all black vertices of the graph at the depot, and artifically setting the distance between any two black vertices equal to a constant λ: $\lambda = \infty$ yields the minimum distance for $|B|$ vehicles, $\lambda = 0$ yields the minimum distance for at most $|B|$ vehicles, and $\lambda = -\infty$ yields the minimum distance for $\lceil |W|/Q \rceil$ vehicles. In the BWTSP, we sometimes refer to Q as the *capacity* of a black-to-black path or chain.

An application of the directed BWTSP arises in short-haul airline operations (Talluri 1998, Mak and Boland 2000). A flight leg between two stations p and q is represented by a white vertex v_{pq}, and a maintenance station s corresponds to a black vertex v_s. An arc (v_i, v_j) is defined whenever a leg-leg, leg-maintenance or maintenance-leg sequence is feasible, and the corresponding value of c_{ij} is the operating cost for the airline of performing the sequence (v_i, v_j). The problem is to determine a flying sequence such that the number of takeoffs and landings, as well as the total operating cost between any two maintenance connections are bounded above. Since this problem is directed, it is often equivalently modeled in terms of arc routing (Talluri 1998). The undirected

case has applications in telecommunications (Wasem 1991, Cosares, Deutsch, Saniee and Wasem 1995). When designing survivable fiber networks using the SONET technology, one seeks a shortest Hamiltonian cycle comprising *hubs* (white vertices) and *ring offices* (black vertices) with upper limits on both the number of links (or hubs) and chain length between consecutive ring offices.

Few algorithms are known for the BWTSP. Mak and Boland (2000) have developed a simulated annealing heuristic for the problem, as well as a lower bounding procedure based on Lagrangean relaxation. They provide computational results on a 36–vertex example. The algorithm proposed by Wasem (1991) determines an optimal solution through implicit enumeration and is therefore suited for small scale or highly constrained instances. This author reports results for networks with up to 240 edges and a capacity $Q = 2$. Bourgeois, Laporte and Semet (2003) have developed and compared three heuristics for the undirected case, based on classical procedures for the TSP.

Our aim is to develop an exact branch-and-cut algorithm for the undirected BWTSP. In Section 1, we present an integer linear programming formulation and several classes of valid inequalities for the problem. The branch-and-cut algorithm, heuristics, separation procedures, and branching strategies are described in Section 2, followed by computational results in Section 3, and by conclusions in Section 4.

1. Formulation and valid inequalities

To model the undirected BWTSP as an integer linear program, define binary variables x_e equal to 1 if and only if edge $e \in E$ appears in the solution. We sometimes use the notation x_{ij} instead of x_e whenever $e = (v_i, v_j)$. Given non-empty subsets S and T of V, let $E(S) = \{(v_i, v_j) : v_i, v_j \in S\}$, $\delta_T(S) = \{(v_i, v_j) : v_i \in S, v_j \in T \setminus S\}$, $\delta(S) = \delta_V(S)$, and $\delta_T(v_j) = \delta_T(\{v_j\})$. Also define $x(E') = \sum_{e \in E'} x_e$ for $E' \subseteq E$. If $S \subseteq V$ and $e = (v_i, v_j) \in E$, let $c_e = c_{ij}$ and

$$\overline{c}_e(S) = \begin{cases} L/2 - c_e & \text{if } e \in \delta_B(S) \\ L/2 - c_e - \min\limits_{e' \in \delta_B(v_j) : v_i \in S} \{c_{e'}\} & \text{if } e \in \delta_W(S). \end{cases}$$

The formulation is then as follows.

$$\text{Minimize } z = \sum_{e \in E} c_e x_e \tag{1}$$

subject to

$$x(\delta(v)) = 2 \qquad (v \in V) \tag{2}$$

$$x(\delta(S)) \geq 2 \qquad (S \subset V, 3 \leq |S| \leq n-3) \tag{3}$$

$$x(\delta(S)) \geq 2 \ \lceil |S|/Q \rceil \quad (S \subseteq W, |S| \geq 2) \tag{4}$$

$$\sum_{e \in E(S)} c_e x_e \leq \sum_{e \in \delta(S)} \bar{c}_e(S) x_e \qquad (S \subseteq W) \tag{5}$$

$$x_e = 0 \text{ or } 1 \qquad (e \in E). \tag{6}$$

In this formulation, constraints (2), (3) and (6) are the standard TSP degree, connectivity and integrality constraints, respectively (Dantzig, Fulkerson and Johnson 1954). If $W = \emptyset$, constraints (4) and (5) do not apply and the BWTSP is then clearly a TSP. Constraints (4) are *cardinality constraints*. They correspond to the VRP connectivity-capacity constraints (Laporte, Nobert and Desrochers 1985) applied to W. Constraints (5) are *length constraints* which impose the maximum length restriction between consecutive black vertices. We now prove their validity.

Proposition 1. Constraints (5) are valid inequalities for the BWTSP.

Proof. We first establish the validity of the following inequality for $S \subseteq W$:

$$\sum_{e \in E(S)} c_e x_e + \sum_{e \in \delta_B(S)} c_e x_e + \sum_{e \in \delta_W(S)} \tilde{c}_e(S) x_e \leq (L/2) \sum_{e \in \delta(S)} x_e, \tag{7}$$

where $e = (v_i, v_j), v_i \in S$ and $\tilde{c}_e(S) = c_e + \min_{e' \in \delta_B(v_j)} \{c_{e'}\}$. The left-hand side of (7) computes a lower bound on the total length of all black-to-black chains passing through S. The first term is the length of all edges of these chains having their two extremities in S. The second term is the length of edges of these chains linking a vertex of S to a black vertex outside S. The third term computes the length of all edges from a vertex of S to a white vertex outside S, plus a lower bound on the length

required to reach a black vertex. The right-hand side of (7) multiplies L by an upper bound on the number of black-to-black chains passing through S. We obtain (5) from (7) by using the definitions of $\tilde{c}_e(S)$ and $\overline{c}_e(S)$. □

The overall model is valid for the BWTSP. Indeed, constraints (2), (3) and (6) are the necessary and sufficient TSP conditions; as in the case of the capacitated VRP (see, e.g., Araque et al., 1994) constraints (4) are necessary and sufficient conditions for the cardinality constraints. The necessity of (5) has been proved in Proposition 1. To see that these constraints are sufficient consider the set S of white vertices in a black-to-black chain contained in a BWTSP solution. For this set S the third term of (7) vanishes, the right-hand side is equal to L, and the left-hand side is the length of the black-to-black chain. In other words, constraints (7) which are algebraically equivalent to (5) are sufficient to eliminate any black-to-black chain longer than L.

Since any BWTSP solution is feasible for the TSP, all valid inequalities for the TSP (see, e.g., Grötschel and Padberg 1985, Naddef and Rinaldi 1993) are also valid for the BWTSP. In the algorithm we will make use of comb inequalities. Further families of valid inequalities specific to the BWTSP can also be derived.

1.1 Cardinality constraints

The first set of valid inequalities are obtained by establishing linear relationships between the feasible values of $x(\delta_B(S))$ and $x(\delta_W(S))$ for any $S \subseteq W$. These constraints are depicted in Figure 1. Letting $k = \lceil |S|/Q \rceil$, we express $|S|$ as $kQ - m$, where $m \in \{0, ..., Q-1\}$ and k is a lower bound on the number of black-to-black chains passing through S. Figure 1 corresponds to the case $|S| = 13, k = 4, Q = 4$, and $m = 3$. We first concentrate on the region defined by $x(\delta_B(S)) \leq 2k$ and we note that constraint (4) corresponds to the straight line passing through $(0, 2k)$ and $(2k, 0)$. A finer analysis of this region is now provided.

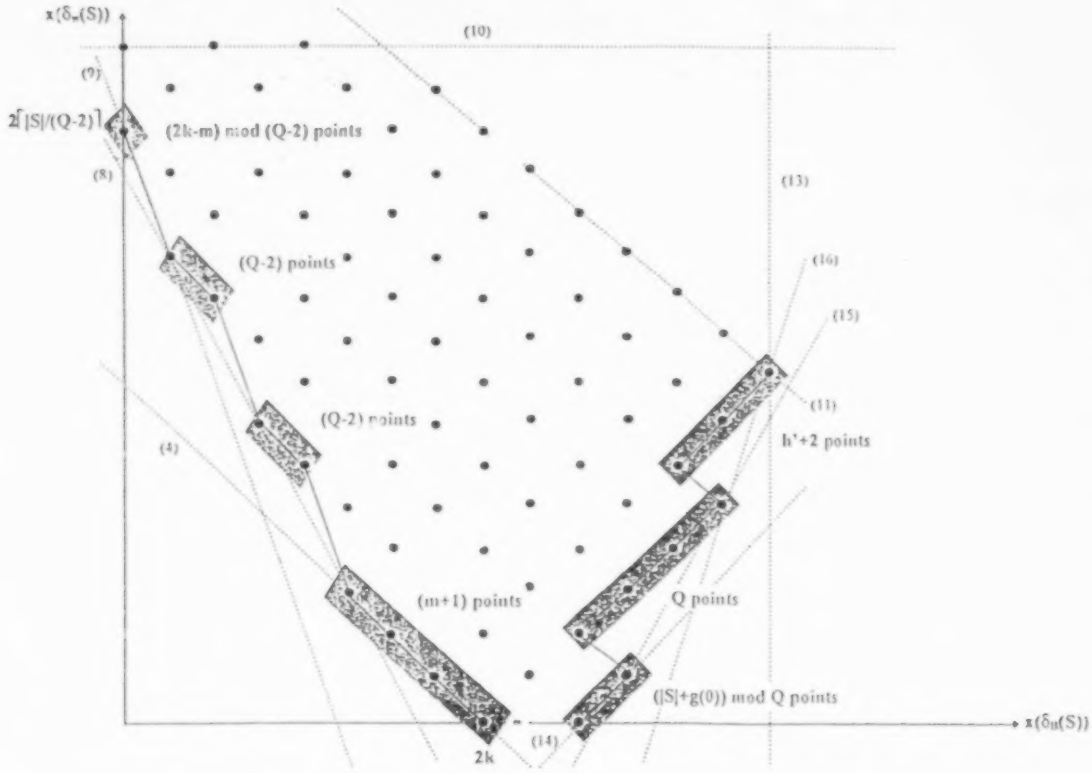

Figure 1 Cardinality constraints for the BWTSP

Proposition 2. If $Q \geq 3$, then for all $S \subset W$, the constraints

$$(Q-2)x(\delta_W(S)) + Qx(\delta_B(S)) \geq 2\,|S| \tag{8}$$

are valid inequalities for BWTSP.

Proof

Inequality (8) can be rewritten as $x(\delta(S)) \geq 2\lceil(|S| + x(\delta_W(S)))/Q\rceil$, where $(|S| + x(\delta_W(S)))$ is a lower bound on the number of white vertices in the chains passing through S □

Lemma 1. For a fixed $\mu \in \{0, ..., 2k\}$ and $S \subset W$, let $\xi = \min_{x(\delta_B(S))=2k-\mu} \{x(\delta_W(S))\}$. If $Q \geq 3$, then $\xi \geq \mu + 2\lceil(\max\{0, \mu - m\})/(Q-2)\rceil$, where $m = kQ - |S|$.

Proof. Given a set $\{i_1, ..., i_r\}$ of white vertices, a chain $(i_0, i_1, ..., i_r, i_{r+1})$ is said to be *black-to-black* (BB) if $i_0, i_{r+1} \in B$, *black-to-white* (BW) if $i_0 \in B, i_{r+1} \in W$, or $i_0 \in W, i_{r+1} \in B$, and *white-to-white* (WW) if $i_0, i_{r+1} \in W$. A BB chain contains at most Q vertices of S, a BW chain contains at most $Q-1$ vertices of S, and a WW chain contains at most $Q-2$ vertices

of S. By (4), at least k chains (BB, BW or WW) are necessary to feasibly insert the vertices of S. This lower bound is based on the assumption that Q vertices of S can be allocated to a chain traversing S. If $x(\delta_B(S)) = 2k - \mu$, then there exists a solution satisfying the cardinality constraints in which the k chains are as follows: $\lfloor (2k-\mu)/2 \rfloor$ BB chains, $(2k-\mu) \bmod (2)$ BW chains, and $\lfloor \mu/2 \rfloor$ WW chains. By (4), the number of vertices of S on the k chains is at most $kQ - [(2k-\mu) \bmod (2) + 2\lfloor \mu/2 \rfloor] = kQ - \mu$. If $m > \mu$, then no additional WW chains are required; otherwise, $\lceil (\mu - m)/(Q-2) \rceil$ additional WW chains are necessary, and the thesis follows.

□

Proposition 3. If $Q \geq 4, k \geq \lceil (m+1)/2 \rceil$, and $f = (2k-m) \bmod (Q-2) \neq 0$ (see Figure 1) then for all $S \subseteq W$, the constraints

$$x(\delta_W(S)) + (1 + 2/f)\, x\,(\delta_B(S)) \geq 2\lceil |S|/(Q-2) \rceil \tag{9}$$

are valid inequalities for BWTSP.

Proof. Observe that (8) and (9) are both satisfied for $x(\delta_B(S)) = f$ and $x(\delta_W(S)) = 2\lceil |S|/(Q-2) \rceil - f - 2$ since f can be rewritten as $(2k-m) - (Q-2)\lfloor (2k-m)/(Q-2) \rfloor$, and $\lceil |S|/(Q-2) \rceil = k + \lceil (2k-m)/(Q-2) \rceil$. Also if $x(\delta_W(S))$ represents the vertical axis and $x(\delta_B(S))$ represents the horizontal axis, then the slope of (9) is $-(1+2/f) \leq -(Q-1)/(Q-3)$, since $f \leq Q-3$, which is smaller than the slope $-Q/(Q-2)$ of (8). Thus (9) is dominated by (8) for $x(\delta_B(S)) \geq f$ and is therefore valid in this range. If $x(\delta_B(S)) \leq f-1$, then $\mu > m$ and, from Lemma 1, we have $\xi \geq \lceil (|S| - x(\delta_B(S))/(Q-2)) \rceil - x(\delta_B(S))$. Since $f = (2k-m) \bmod (Q-2) = |S| \bmod (Q-2)$ and $x(\delta_B(S)) \leq f-1$, we obtain $\xi \geq 2\lceil (|S|/(Q-2)) \rceil - x(\delta_B(S))$. Hence (9) is also valid for $x(\delta_B(S)) \leq f-1$. □

The following constraints describe the top and right-hand side parts of the feasible region depicted in Figure 1, defined by $x(\delta_B(S)) \geq 2k$.

Proposition 4. For all $S \subset W$, the constraints

$$x(\delta_W(S)) \leq 2 \min \{|S|, |W \setminus S|\} \tag{10}$$

are valid inequalities for BWTSP.

Proof. Follows from the definition of $x(\delta_W(S))$ and from (2). □

Proposition 5. For all $S \subset W$, the constraints

$$x(\delta(S)) = x(\delta_W(S)) + x(\delta_B(S)) \leq 2 \text{ min } \{|S|, |V \setminus S|\} \tag{11}$$

are valid inequalities for BWTSP.

Proof. Follows from the definition of $x(\delta(S))$ and from (2).

In Propositions 6 to 9, let

$$g(N) = (|B| - N)Q - |W|. \tag{12}$$

The value of $g(N)$ represents the total unused capacity of all black-to-black chains containing at least one white vertex if the solution contains N black-to-black edges.

Proposition 6. If $Q \geq 2$, then for all $S \subset W$, the constraints

$$x(\delta_B(S)) \leq \min \{2|B|, \lfloor g(0)/(Q-1) \rfloor + |S|\} \tag{13}$$

are valid inequalities for BWTSP.

Proof. The inequality $x(\delta_B(S)) \leq 2|B|$ follows from (2). In order to prove that $x(\delta_B(S)) \leq \lfloor g(0)/(Q-1) \rfloor + |S|$, define m_{BB} and m_{BW} as the number of BB chains and BW chains passing through S in a feasible solution. Moreover, let S_{BB} and S_{BW} be the vertices of S along these chains. The total unused capacity of BB chains passing through S is equal to $Qm_{BB} - |S_{BB}|$, which is at most equal to $Qm_{BB} - m_{BB} \leq g(0)$. Therefore $m_{BB} \leq \lfloor g(0)/(Q-1) \rfloor$. Furthermore, $m_{BW} \leq |S_{BW}| \leq |S| - |S_{BB}| \leq |S| - m_{BB}$. Hence, $x(\delta_B(S)) = 2\, m_{BB} + m_{BW} \leq m_{BB} + |S| \leq \lfloor g(0)/(Q-1) \rfloor + |S|$. □

Moreover, it can be easily shown that the upper bound in (13) is attained if $|W \setminus S| \geq |S| - \lfloor g(0)/(Q-1) \rfloor$. To develop further valid inequalities, we use the following lemma.

Lemma 2. For $S \subset W$, let $x(\delta_W(S)) = \eta$. Then $x(\delta_B(S)) \leq \eta + 2\lfloor (g(x(E(B))) + |S| - \eta)/Q) \rfloor$ for $\eta = 0, ..., 2\min\{|S|, |W \setminus S|\}$.

Proof. Using the same arguments as in the proof of Proposition 6, it is easy to show that $m_{BB} \leq \lfloor (g(x(E(B))) + |S| - m_{BW})/Q \rfloor$. Hence $x(\delta_B(S)) = 2m_{BB} + m_{BW} \leq m_{BW} + 2\lfloor (g(x(E(B))) + |S| - m_{BW})/Q \rfloor \leq \eta + 2\lfloor (g(x(E(B))) + |S| - \eta)/Q \rfloor$. □

Proposition 7. If $Q \geq 3$, then for all $S \subset W$, the constraints

$$x(\delta_B(S)) - x(\delta_W(S)) \leq 2\lfloor (g(0) + |S|)/Q \rfloor \tag{14}$$

are valid inequalities for BWTSP.

Proof. The validity of inequality (14) follows from Lemma 2 by observing that $\lfloor (g(x(E(B))) + |S| - \eta)/Q \rfloor \leq \lfloor (g(x(E(B))) + |S|)/Q \rfloor \leq \lfloor (g(0) + |S|)/Q \rfloor$. □

Proposition 8. If $Q \geq 3$, then for all $S \subset W$, the constraints

$$Qx(\delta_B(S)) - (Q-2)x(\delta_W(S)) \leq 2(g(x(E(B))) + |S|) \tag{15}$$

are valid inequalities for BWTSP.

Proof. Lemma 2 implies $x(\delta_B(S)) \leq \eta + 2(g(x(E(B))) + |S| - \eta)/Q$. The thesis follows by substituting η by $x(\delta_W(S))$ and rearranging this expression. □

Proposition 9. Let $h = \lfloor g(0)/(Q-1) \rfloor + (Q-2)\lfloor (g(0) + |S|)/Q \rfloor - g(0)$ and $h' = h \bmod (Q-2)$. If $h \geq 1$, $h' \geq 1$ and $Q \geq 3$, then for all $S \subset W$ the constraints

$$h'x(\delta_W(S)) - (h'+2)x(\delta_B(S)) \geq -2|S| - 2(h'+1)\lfloor g(0)/(Q-1) \rfloor \tag{16}$$

are valid inequalities for BWTSP. Moreover, for $x(\delta_B(S)) \geq |S| + \lfloor g(0)/(Q-1) \rfloor - h'$ (16) dominates (15).

Proof. First, we prove that (16) is valid. Since $x(\delta_B(S)) = 2m_{BB} + m_{BW} \leq 2\lfloor g(0)/(Q-1) \rfloor + x(\delta_W(S))$ and $x(\delta_B(S)) \leq \lfloor g(0)/(Q-1) \rfloor + |S|$ (see (13)), it follows that, (16) is valid. Now, both (15) and (16) are satisfied as equalities for $x(\delta_B(S)) = |S| + \lfloor g(0)/(Q-1) \rfloor - h'$ and $x(\delta_W(S)) = |S| - \lfloor g(0)/(Q-1) \rfloor - h' - 2$. Since the slope of (16) is larger than the slope of (15), it follows that (16) dominates (15) for $x(\delta_B(S)) \geq |S| + \lfloor g(0)/(Q-1) \rfloor - h'$. □

Under their respective hypotheses, inequalities (8) (10), (11), and (16) are satisfied as equalities by at least two feasible solutions whenever $L = \infty$. Therefore, these inequalities, along with the trivial relations $x(\delta_B(S)) \geq 0$ and $x(\delta_W(S)) \geq 0$, are facet defining for the convex hull of the projection of the BWTSP feasible solutions on the $(x(\delta_B(S)), x(\delta_W(S))$ two-dimensional space.

1.2 Length constraints

We now derive valid inequalities related to the maximal length of black-to-black chains.

Proposition 10. Let $S \subseteq W$, and v_p, v_q be two given vertices of $B(p \neq q)$, and denote by $H_S(p,q)$ the length of a shortest Hamiltonian chain between v_p and v_q including all vertices of S. If $H_S(p,q) > L$, then the constraint

$$x_{pr} + x_{sq} + x(E(S)) \leq |S| \tag{17}$$

is a valid inequality for BWTSP for any $v_r, v_s \in S(r \neq s)$. Furthermore, if $H_S(p,q) > L$ for all choices of two vertices $v_p, v_q \in B\,(p \neq q)$, then (17) can be strengthened to

$$x(\delta(S)) \geq 4. \tag{18}$$

Proof. Since $H_S\,(p,q) > L$, no solution can contain a chain between v_p and v_q and passing through all vertices of S. If a solution contained such a chain, the left-hand side of (17) would be equal to $|S| + 1$. The second condition means that no black-to-black chain passing through all vertices of S can feasibly exist. If there were one, the left-hand side of (18) would be equal to 2. Conversely, given a set $S \subset W$, two vertices $v_k, v_l \in B(v_k \neq v_l)$ and two vertices $v_r, v_s \in S(v_r \neq v_s)$, for any feasible solution either the hypotheses of (17) do not hold or the left hand-side of (17) is less than or equal to $|S|$. Therefore, no feasible solution is eliminated by (17). □

Constraints (17) are referred to as *Hamiltonian chain inequalities*. They are dominated by constraints (4) whenever $|S| > Q$.

Proposition 11. Consider two disjoint subsets $S \subseteq W$ and $T \subseteq V$, with $2 \leq |S| \leq Q$, and $v_p \in B \backslash T$. If $H_S(p,q) + \min_{v_j \in B \backslash \{v_p\}} \{c_{qj}\} > L$ for all $v_q \in T$, then the constraint

$$2x_{pr} + 2x(E(S)) + \sum_{v_q \in T} x_{sq} \leq 2|S| \tag{19}$$

is a valid inequality for BWTSP for any $v_r, v_s \in S$ $(r \neq s)$.

Proof. First we observe that $\min_{v_j \in B \backslash \{v_k\}} c_{qj} = 0$ whenever $v_q \in T \cap B$. Moreover, $H_S(p,q) + \min_{v_j \in B \backslash \{v_K\}} c_{qj}$ is a lower bound on the length of any path from v_p to another black vertex, passing through all vertices of S and possibly a white vertex of T. Let $t = \sum_{v_q \in T} x_{sq}$. In a feasible solution, only three cases are possible: $t = 0$ and $x_{pr} + x(E(S)) \leq |S|$, $t = 1$ or 2 and $x_{pr} + x(E(S)) \leq |S| - 1$ in order to prevent a length constraint violation. In all cases (19) holds. □

Constraints (19) are refereed to as *single-mop inequalities*. They are clearly dominated by the Hamiltonian chain inequalities (17) whenever $|T| = 2$. There exist, however, situations where (19) is violated whereas (17) is satisfied. Consider for example the situation depicted in Figure 2. (In Figures 2, 3 and 4, the x_e values are shown on theedges. Some edges needed to satisfy the degree constraints are not shown). It corresponds to a violated single–mop inequality, while (17) is not violated for any $v_q \in T$.

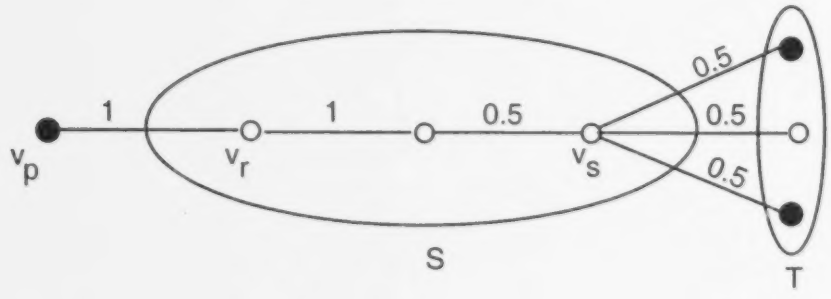

Figure 2 Example showing a violation of the single-mop inequality, but no violation of the Hamiltonian chain inequality.

We now introduce the *double-mop inequalities*.

Proposition 12. Consider three mutually disjoint subsets $S \subseteq W, T_1, T_2 \subseteq V$, with $3 \leq |S| \leq Q, |T_1| \geq 2$ and $|T_2| \geq 2$. If $H_S(p,q) + \min_{v_i \in B \setminus \{v_q\}} \{c_{pi}\} + \min_{v_j \in B \setminus \{v_p\}} \{c_{qj}\} > L$ for all $v_p \in T_1, v_q \in T_2$, then the constraints

$$2x(E(S)) + \sum_{v_p \in T_1} x_{pr} + \sum_{v_q \in T_2} x_{sq} \leq 2|S| - 1 \tag{20}$$

are valid inequalities for BWTSP for any $v_r, v_s \in S (r \neq s)$.

Proof. The proof is similar to that of Proposition 11. Let $t = \sum_{v_p \in T_1} x_{pr} + \sum_{v_q \in T_2} x_{sq}$. In a feasible solution, only five cases are possible: $t = 0$ or 1 and $x(E(S)) \leq |S| - 1$, $t = 2$ or 3 and $x(E(S)) \leq |S| - 2, t = 4$ and $x(E(S)) \leq |S| - 3$. In all cases (20) holds. □

It is interesting to observe that the double-mop inequality (20) dominates the single-mop inequality (19) since $2x_{pr} \leq \sum_{v_j \in T_1} x_{ij} + 1$ for all $v_p \in T_1$, but single-mop inequalities are easier to separate and are still useful. However, there is no dominance relationship between Hamiltonian chain inequalities (17) and double-mop inequalities (20). The example depicted in Figure 3 corresponds to a case where (17) is satisfied for all $v_p \in T_1, v_q \in T_2$, but (20) is violated. Figure 4 illustrates a situation were the double-mop inequality (20) is always satisfied, but the Hamiltonian chain inequality associated with v_p and v_q is violated.

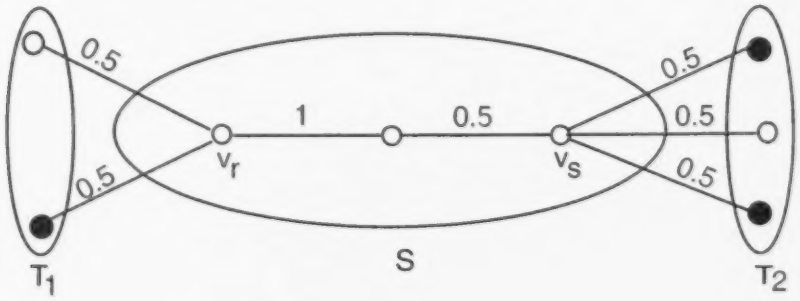

Figure 3 Example showing a violation of the double-mop inequality (20), but no violation of the Hamiltonian chain inequality.

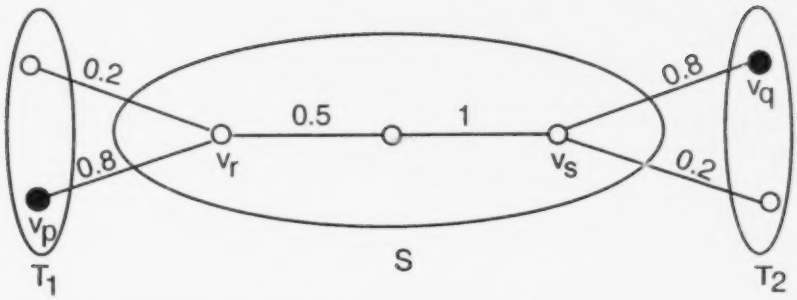

Figure 4 Example showing a violation of the Hamiltonian chain inequality, but no violation of the double-mop inequality (20).

2. The branch-and-cut algorithm

We have developed the following branch-and-cut algorithm for BWTSP. For an overview and bibliography on this approach, see Caprara and Fischetti (1997).

Step 1 (Initial upper bound). Attempt to compute an initial upper bound $\overline{z}$ on the optimal value z^* by means of a heuristic. If no feasible solution can be obtained, set $\overline{z} := L|B| + 1$.

Step 2 (First node of the search tree). Define a first subproblem by the linear program containing all degree constraints (2), one connectivity-capacity constraint (4) for $S = W$, and $0 \leq x_e \leq 1$ for all $e \in E$. Insert this subproblem in a list.

Step 3 (Termination test). If the list is empty, stop with an infeasible solution or with the best known feasible solution (which is then the optimum). Otherwise, select a subproblem from the list according to the most promising branch (or least lower bound) strategy.

Step 4 (Subproblem solution). Solve the linear program associated with the subproblem and let z be the solution value. If the subproblem is infeasible or $z \geq \overline{z}$, go to Step 3. If the solution is feasible for BWTSP, set $\overline{z} := z$, record the solution, and go to Step 3.

Step 5 (Constraint elimination). Among all constraints introduced after Step 2, eliminate those that have been inactive for five executions of Step 4.

Step 6 (Constraint generation). Identify violated inequalities considering, in this order, constraints of type (3), (4), (8), (9), (14), (15), (16), (5), (17), (18), (19), (20), and comb inequalities. If no

inequality can be generated, go to Step 7. Otherwise, generate all identified inequalities violated and go to Step 4.

Step 7 (Upper bound update). Attempt to generate feasible solutions using the *support graph* induced by all edges e for which the value of x_e is positive. If any of these solutions yields an improved upper bound value, update $\overline{z}$.

Step 8 (Branching). Create two subproblems by branching on a fractional variable or on a violated constraint, insert these in the list, and go to Step 3.

We now comment on some specific steps of this algorithm.

2.1 Step 1: Initial upper bound

An attempt is made to obtain a first feasible solution by means of "Algorithm C1-F-I" developed by Bourgeois, Laporte and Semet (2003), which we now outline.

i) Determine a feasible TSP solution on G by means of the GENIUS heuristic (Gendreau, Hertz and Laporte 1992) with a neighbourhood size equal to six. If this solution is feasible for BWTSP, go to iii).

ii) Attempt to make the current solution feasible for BWTSP by performing the following operations:

   - Consider the two end vertices of a black-to-black chain. If the distance between these two vertices exceeds L, identify another black vertex resulting in the least length increase when optimally located in the chain. Remove it from its current location and insert it in the chain, thus creating two new black-to-black chains. Repeat until all black-to-black chains have been considered.
   - If the current length of a black-to-black chain exceeds L, or if its number of white vertices exceeds Q, remove a white vertex from this chain and reinsert it in another black-to-black chain whose length is less than L or containing at most $Q - 1$ vertices. Repeat until all black-to-black chains have been considered.
   - If no feasible solution has been identified, stop.

iii) Attempt to improve the current feasible solution by means of a 2–opt algorithm adapted to the BWTSP, where an edge exchange is only implemented if it does not create an infeasibility.

2.2 Step 6: Separation techniques

We have used five different separation techniques for the detection of violated constraints in Step 6.

The maximum tree and maximum flow techniques

As suggested by Fischetti, Salazar and Toth (1997), first construct a maximum spanning tree with respect to the current x_e values on the support graph of the current solution. Then, by successively removing each edge of the spanning tree, one can readily verify whether either of the two subtrees yields a violated connectivity constraint (3). If the heuristic fails to detect a constraint violation, an exact maximum flow algorithm is applied (Padberg and Grötschel, 1985).

The white forest technique

First remove from the support graph all edges incident to at least one black vertex. Then apply to each connected component (containing only white vertices) the maximum spanning tree technique to identify candidate sets S yielding violated constraints of type (4), (5), (8), (9), (14), (15) and (16). For constraints (14) and (15), the value of $x(E(B))$ does not depend on S and is available before computing the spanning tree. It is worth observing that constraints (8) could also be separated exactly in $O(n^4)$ time by solving a maximum flow problem (the proof is provided in the Appendix), but this procedure turns out to be rather time consuming and is not applied here.

The tabu search technique

Augerat et al. (1998) have used tabu search to identify violated VRP capacity constraints. We have extended their technique to detect violated inequalities (4), (8) and (9) for the BWTSP. These constraints can be generically expressed as a function $\theta(S) \geq 0 \, (S \subseteq W)$. Tabu search is used in the hope of identifying one or several sets S for which $\theta(S) < 0$. In the case of capacity constraints, it has already been observed by Augerat et al. that such sets S are likely to have a cardinality in

the vicinity of $Q, 2Q, 3Q, ...$ We have used this idea to develop a multistart tabu search (Glover and Laguna 1997) from these breakpoints. The search starts with randomly constituted sets of white vertices having a given cardinality, and neighbour solutions are obtained by removing and including vertices from these sets while ensuring that their cardinality remains within an acceptable interval. As is commonly done in some tabu search implementations (see, e.g., Taillard 1991, Gendreau, Hertz and Laporte 1994), we have used random tabu tenures. The search stops after 100 iterations.

This separation technique is of course time consuming as compared with the previous approaches and we have used it only when no violated constraints could be identified with the first two techniques. Moreover, it was only applied to constraints (4), (8) and (9) which turned out to be the most useful. We have employed the tabu search technique first for constraints (4) and then for constraints (8). However, whenever a violation of (8) was identified, a check was made to see whether a stronger constraint of type (9) for the same set S could be found. In such a case, only constraint (9) was generated.

The Lagrangean relaxation technique

The following technique is applied to identify violated constraints involving the computation of a shortest Hamiltonian chain. These are constraints (17), (18), (19) and (20) in our case. First compute on the support graph a maximum spanning tree with respect to the x_e values. Then attempt to identify structures yielding a violated constraint. In the case of the Hamiltonian chain inequalities, we simply seek black-to-black chains in the spanning tree whose intermediary vertices are all white. In the case of single-mop inequalities, we seek black-to-white chains containing a single black vertex at one end and branching into at least three edges from the other end vertex. In the case of double-mop inequalities, we seek chains of white vertices branching into at least three edges at each end. Once a constraint violating structure has been identified, one must check whether the conditions of Proposition (10), (11) or (12) are satisfied. In all cases, this amounts to checking whether the length $H_S(p, q)$ of a shortest Hamiltonian chain exceeds a certain threshold ρ. (The

value of ρ is equal to L in constraints (17) and (18), to $L - \min_{v_j \in B \setminus \{v_p\}} \{c_{qj}\}$ in constraints (19), and to $L - \min_{v_i \in B \setminus \{v_q\}} \{c_{pi}\} - \min_{v_j \in B \setminus \{v_p\}} \{c_{qj}\}$ in constraints (20)). These chains are computed with respect to the c_{ij} coefficients. They link vertices v_p and v_q and go through all vertices of S. A first check is made by computing a lower bound on $H_S(p,q)$ by means of a shortest spanning tree on the subgraph induced by $\{v_p, v_q\} \cup S$. If the length of the tree exceeds ρ, a violated constraint has been identified. If it is less than 0.6ρ, there is little likehood that a violated constraint using this particular structure will be found, and the search is abandonned. Otherwise, an additional effort to increase the lower bound is made by applying the classical Lagrangean relaxation procedure first proposed by Held and Karp (1971) in the context of the undirected TSP, using an implementation similar to that of Volgenant and Jonker (1982). We allowed a maximum of 20 ascent iterations in order for the lower bound to exceed ρ. When this is possible, a violated constraint has been identified; otherwise the search on structure is abandoned.

The block technique

The comb inequalities are generated as in Padberg and Rinaldi (1990). In an undirected graph, deleting a vertex and all its incident edges may disconnect the graph. Any such vertex is called an *articulation vertex*. A *block* is a maximal subgraph containing no articulation vertex. In the Padberg and Rinaldi procedure, all blocks of the graph induced by the positive x_{ij} variables different from 1 are first identified, and each block is then considered as a candidate for the "handle" of the comb, other blocks are considered as potential "tooth sets".

2.3 Step 7: Upper bound update

As in Fischetti and Toth (1997) we attempt to generate feasible solutions using the support graph. This is done by implicit enumeration starting with an arbitrary black vertex, and following untraversed edges e for which the value of x_e is positive. We set an upper limit of 100 + 10 $|V|$ on the number of backtracking steps, but this limit was seldom reached in our tests. This technique proved rather useful on tightly constrained instances.

2.4 Step 8: Branching

We have implemented two branching strategies. The first strategy consists of branching on a fractional variable x_e having the highest *score* defined as $c_e \min\{x_e, 1 - x_e\}$. This favours variables having a large cost and whose current value is close to 0.5. The second strategy consists of branching on a constraint of type (4) whose left-hand side is close to an odd number in the current solution. We found that the following rule works well in practice: among the sets $S \subseteq W$ identified when trying to generate violated cardinality constraints (4), select one for which $2.9 \leq x(\delta(S)) \leq 3.1$ and having maximum cardinality, and create the two branches $x(\delta(S)) = 2$ and $x(\delta(S)) \geq 4$ (since $x(\delta(S)) = 3$ cannot occur). If such a set cannot be identified, branching is made on a variable. Because branching occurs after no violated constraints can be identified, the cardinality of S will not exceed Q and this branching will either force all vertices of S to be on the same black-to-black chain, or to belong to at least two chains.

3. Computational results

The branch-and-cut algorithm just described was coded in C and run on a Pentium II processor clocked at 500 MHz with 512 Mbytes RAM. We used CPLEX 8.0 for the solution of the subproblems.

3.1 Instance generation

The algorithm was tested on 540 randomly generated instances and on 22 TSPLIB instances (Reinelt, 1991). The random instances were generated as follows, using three user-controlled parameters α, β and γ.

Step 1. Generate $n = |V|$ integer coordinate points in the $[0, 1000]^2$ square according to a discrete uniform distribution. Compute edge costs as Euclidean distances plus 0.5, rounded down to the nearest integer.

Step 2. Set $Q := \alpha n$

Step 3. Set $|B| := \beta b_{\min}$, where $b_{\min}$ is the minimum number of vertices required to ensure feasibility. It is equal to the smallest integer satisfying $b_{\min} - \lceil (n - b_{\min})/Q \rceil \geq 0$. Designate the first $|B|$ vertices as black.

Step 4. Set $L := \gamma \ell_{\max}$, where $\ell_{\max}$ is the longest path length between two consecutive black vertices when the BWTSP is solved to optimality with $L = \infty$.

We have generated five instances for each combination of the following values: $n = |V|$ =20, 40, 60, 80, 100; α =0.2, 0.35, 0.5; β =1, 1.33, 1.67; γ = 0.85, 0.95, ∞. Instances with $\gamma < 0.85$ tended to be infeasible and were not fully tested.

We solved the TSPLIB instances with $14 \leq n \leq 100$ (the names are provided in Table 5) for $\alpha = 0.2, 0.35, 0.5; \beta = 1, 1.33, 1.67, \gamma = \infty$ (tests performed with $\gamma = 0.85$ and 0.95 yielded a very low success rate and their results are not presented).

3.2 Test results

We now summarize our test results in five tables whose headings are defined as follows:

α	:	control parameter for Q.
β	:	control parameter for $\|B\|$.
γ	:	control parameter for L.
n	:	number of vertices.
SUCC	:	number of successful instances out of five (if at least three).
FEAS	:	number of feasible instances.
CO	:	number of connectivity constraints (3).
C4	:	number of cardinality constraints (4).
C8	:	number of cardinality constraints (8).
C9	:	number of cardinality constraints (9).
C14	:	number of cardinality constraints (14).
C15	:	number of cardinality constraints (15).
C16	:	number of cardinality constraints (16).
LC	:	number of length constraints (5).
HC1	:	number of Hamiltonian chain inequalities (17).
HC2	:	number of Hamiltonian chain inequalities (18).
SM	:	number of single-mop inequalities (19).
DM	:	number of double-mop inequalities (20).

CM : number of comb inequalities.

LB/z^* : lower bound value obtained at the root node, before any branching, divided by the optimal value z^*. This statistic is an average computed over the number of feasible instances.

NODES : number of nodes in the branch-and-cut tree.

SEC : computing time in seconds.

With the exception of LB/z^*, all statistics are rounded to the nearest integer. We report average results over the number of "successful" instances (out of five), i.e., those for which infeasibility or optimality could be proved within 10,000 seconds. When less than three instances of a given type are successful, no result is reported (apart for one exception in Table 1). Some values of γ yield infeasible instances, but this can only be discovered after the completion of the branch-and-cut search, not when the instance is generated.

In a first series of tests, we compared the two branching strategies (branching on a variable x_e and branching on a constraint of type (4)) for several values of the parameters. The second branching turned to be superior to the first in that it resulted in a reduction of around 50% in the average number of nodes of the search tree and of around 20% in the average computing time. It allowed us to solve some difficult instances that were intractable under the first branching strategy. To illustrate, we report in Table 1 comparative results for $\beta = 1, \gamma = 0.95$, and all values of α and n.

All subsequent tests were carried out with the second branching strategy. The results obtained on randomly generated instances are summarized in Tables 2, 3, 4, corresponding to γ =0.85, 0.95 and ∞, respectively. Tables 2 to 4 show that instances involving up to 100 vertices can be solved within the prescribed computing time. Problem difficulty increases sharply with n, as measured by the SEC and SUCC columns, for example. Larger instances tend to have a lower LB/z^* ratio and give rise to the generation of more constraints. Tighter instances (corresponding to low values of α, β or γ) are also more difficult. In particular, γ is a major derminant in problem difficulty. The generation mode used for γ was designed to make the length constraint binding for $\gamma < 1$ and this resulted in several infeasible cases requiring an extensive search. For $\gamma = 0.85$ and small values of α and β, several

instances were found to be infeasible without generating any length constraint since a subproblem with a solution value exceeding $|B|L$ can be immediately fathomed. This phenomenon occurs more frequently for large n since then all black-to-black chains tend to have similar cardinalities and lengths. Instances generated with $\gamma = \infty$ tend to be the easiest and always yield an LB/z^* ratio at least equal to 98.4%; when $\gamma = 0.85$ or 0.95, the lowest ratio is 93.8% and 95.6%, respectively. As a rule the number of cardinality constraints of type (4), (8) and (9) tends to be larger when α is small, while more constraints of type (14), (15) and (16) are generated when β is small. Among the various length constraints, more constraints of type (5) are generated because there are considered first and are easier to separate than the other length constraints. Results obtained on TSPLIB instances are presented in Table 5. These instances seem slightly more difficult than those that were generated randomly in the sense that we could solve only a limited number for $\gamma = 0.85$ and 0.95. However, the difficulty of both instance types is comparable for $\gamma = \infty$. We have performed exhaustive tests for $\alpha = 0.2$, 0.35, 0.5, and $\beta = 1.00$, 1.33 and 1.67. For the sake of conciseness we only report results for $\alpha = 0.35$ and $\beta = 1.33$ which are representative values. Four of the TSPLIB instances (st70, gr96, kroA100, and kroB100) could not be solved within 10,000 seconds for these parameter values.

Table 1 Comparison between the two branching strategies

for $\beta = 1$ and $\gamma = 0.95$ on randomly generated instances

α	n	Branching on a variable			Branching on a constraint		
		SUCC	NODES	SEC	SUCC	NODES	SEC
0.2	20	5	389	67	5	594	150
	40	4	2048	1832	4	380	319
	60	2	221	407	4	1470	4136
0.35	20	5	27	2	5	19	2
	40	5	2062	1354	4	913	681
	60	4	934	992	4	335	813
0.5	20	5	144	11	5	67	6
	40	5	85	19	5	39	13
	60	4	6	4	4	8	5
	80	3	731	1307	3	558	2548

Table 2 Summary of computational results for $\gamma = 0.85$ on randomly generated instances

α	β	n	SUCC	FEAS	C0	C4	C8	C9	C14	C15	C16	LC	HC1	HC2	SM	DM	2M	LB/z*	NODES	SEC
0.2	1	20	4	1	173	767	814	42	204	120	0	228	26	4	1	6	15	0.993	90	17
	1.33	20	5	2	55	57	67	16	3	5	0	79	12	5	2	8	6	0.991	23	3
		40	4	3	483	602	4	156	10	9	0	613	54	41	18	38	195	0.963	191	125
	1.67	20	5	2	71	63	64	17	2	2	0	82	16	4	4	23	14	0.984	30	3
		40	5	5	2296	935	2	450	9	6	0	3006	295	284	76	129	792	0.956	376	334
		60	3	3	432	522	0	165	3	2	0	217	12	7	2	3	249	0.979	324	365
0.35	1	20	5	3	83	188	0	78	36	5	1	141	25	15	10	1	31	0.977	302	444
		40	3	1	348	943	0	297	266	61	7	620	13	16	0	1	220	0.955	272	216
		60	5	0	306	607	0	43	246	52	2	672	11	12	3	0	132	- -	156	212
		80	3	1	1815	6665	0	473	2080	470	61	14	0	0	0	0	1141	0.977	402	2520
	1.33	20	5	3	83	188	0	78	36	5	1	141	25	15	10	1	31	0.977	302	444
		40	3	1	348	943	0	300	266	61	7	622	13	16	0	1	220	0.955	272	216
		60	5	0	306	607	0	43	246	52	2	672	11	12	3	0	132	- -	156	211
		80	3	0	57	18	0	1	10	7	2	23	0	0	0	0	66	- -	1	2
	1.67	20	5	5	60	20	0	13	1	0	0	76	19	11	2	3	13	0.955	75	6
		40	4	4	112	72	0	20	1	1	0	90	9	6	2	2	79	0.982	53	18
		60	5	5	2293	1950	0	159	15	8	0	2417	103	113	27	34	2084	0.972	1071	2316
		80	4	4	714	341	0	11	6	4	0	665	47	49	11	8	964	0.987	500	2311
0.5	1	20	5	1	22	23	0	12	18	6	1	37	2	2	0	0	13	0.938	8	1
		40	5	1	45	140	0	17	24	13	0	139	1	0	0	0	47	0.976	17	8
		60	4	0	6	0	0	0	0	0	0	0	0	0	0	0	15	- -	1	1
		80	5	0	56	48	0	2	33	7	1	29	0	0	0	0	69	- -	20	26
	1.33	20	5	1	22	23	0	12	18	6	1	37	2	2	0	0	13	0.938	8	1
		40	5	1	45	140	0	17	24	13	0	139	1	0	0	0	47	0.976	17	8
		60	4	0	14	0	0	0	0	0	0	0	0	0	0	0	19	- -	1	1
		80	5	0	56	48	0	2	33	7	1	29	0	0	0	0	69	- -	20	26
	1.67	20	5	5	8758	5434	0	735	154	93	14	13724	2433	1317	749	213	902	0.940	3686	690
		40	4	3	166	286	0	25	8	4	1	275	13	9	2	3	127	0.969	127	58
		60	5	4	1448	1211	0	42	67	26	7	1864	75	75	11	5	1290	0.972	698	3023

Table 3 Summary of computational results for $\gamma = 0.95$ on randomly generated instances

α	β	n	SUCC	FEAS	C0	C4	C8	C9	C14	C15	C16	LC	HC1	HC2	SM	DM	2M	LB/z*	NODES	SEC
0.2	1	20	5	4	2724	7545	7146	556	3158	1879	0	3764	217	41	26	201	262	0.954	594	150
		40	4	4	2098	3934	415	3985	685	153	2	2227	124	43	48	23	422	0.968	380	319
		60	4	4	9016	29291	0	12452	7428	1694	140	11524	234	236	28	87	2896	0.969	1470	4136
	1.33	20	5	4	43	46	95	16	2	3	0	26	5	1	1	0	4	0.985	25	2
		40	4	4	2909	5306	304	3805	47	47	0	2417	200	51	71	35	567	0.975	484	464
		60	4	4	4882	14788	0	5030	204	154	1	3354	42	31	51	29	2489	0.971	1083	3257
	1.67	20	5	5	48	74	157	27	1	3	0	48	7	4	1	2	10	0.978	27	3
		40	5	5	179	83	1	68	0	0	0	110	18	10	4	5	100	0.978	113	44
		60	5	5	532	713	0	198	2	2	0	280	6	3	1	3	321	0.978	251	373
		80	4	4	822	1047	0	164	18	12	0	775	37	32	18	7	539	0.985	227	826
0.35	1	20	5	5	36	64	0	81	11	3	1	50	4	1	0	0	13	0.989	19	2
		40	4	4	3552	16150	0	1551	2718	773	118	6566	285	92	99	106	770	0.972	913	681
		60	4	3	373	953	0	114	352	132	16	866	57	37	6	3	190	0.979	335	813
	1.33	20	5	5	36	64	0	81	11	3	1	50	4	1	0	0	13	0.989	19	2
		40	4	5	56	176	0	23	34	16	1	101	0	0	0	0	59	0.988	21	10
		60	4	2	373	960	0	114	351	132	16	861	57	37	6	3	188	0.979	335	804
	1.67	20	5	5	24	13	0	16	0	0	0	24	4	2	1	0	12	0.989	12	1
		40	5	5	75	43	0	15	1	0	0	31	3	3	2	3	59	0.993	19	7
		60	5	4	312	372	0	61	3	4	0	278	10	10	2	3	345	0.979	301	671
		80	5	5	751	721	0	52	19	17	0	628	11	11	5	1	933	0.988	247	779
0.5	1	20	5	3	57	164	0	69	63	22	3	160	14	3	7	2	13	0.966	67	6
		40	5	2	62	215	0	26	46	22	1	147	1	0	0	0	52	0.976	39	13
		60	4	0	69	42	0	3	12	7	1	52	1	1	0	0	65	- -	8	5
		80	3	3	1419	1966	0	108	1271	215	38	1004	15	2	1	2	1429	0.984	558	2548
	1.33	20	5	3	57	164	0	69	63	22	3	160	14	3	7	2	13	0.966	67	8
		40	4	5	99	203	0	68	61	19	4	79	1	1	0	1	71	0.987	18	11
		60	4	0	69	42	0	3	12	7	1	52	1	1	0	0	65	- -	8	5
		80	4	3	1419	1966	0	108	1271	215	38	1004	15	2	1	2	1429	0.984	558	2532
	1.67	20	5	5	48	54	0	27	5	4	0	90	16	6	1	4	16	0.962	55	3
		40	5	5	1563	2038	0	153	105	76	13	1913	88	63	25	27	631	0.974	453	995
		60	4	4	190	174	0	10	6	5	1	149	6	6	1	0	189	0.988	114	236

Table 4 Summary of computational results for $\gamma = \infty$ on randomly generated instances

α	β	n	SUCC	C0	C4	C8	C9	C14	C15	C16	2M	LB/z*	NODES	SEC
0.2	1	20	5	26	41	58	11	19	14	0	6	0.998	3	0
		40	5	109	241	37	263	41	8	0	56	0.985	43	18
		60	5	2557	9545	0	5149	2655	604	64	886	0.984	331	787
	1.33	20	5	13	15	15	7	2	3	0	3	0.999	2	1
		40	5	90	144	7	112	1	1	0	41	0.991	27	10
		60	5	1109	3226	0	1189	22	18	0	723	0.983	191	327
	1.67	20	5	27	19	31	9	0	0	0	12	0.999	2	0
		40	5	69	35	0	99	0	0	0	53	0.995	9	3
		60	5	264	286	0	98	1	1	0	174	0.984	67	55
		80	5	341	586	0	139	3	2	0	244	0.990	97	225
0.35	1	20	5	23	31	0	87	9	2	1	14	0.998	3	0
		40	5	83	224	0	98	53	19	4	60	0.990	25	8
		60	5	198	555	0	97	155	39	5	144	0.985	58	66
		80	4	471	1256	0	86	508	117	29	450	0.985	107	327
	1.33	20	5	23	31	0	87	9	2	1	14	0.998	3	1
		40	5	83	224	0	98	53	19	4	60	0.990	25	8
		60	5	198	555	0	97	155	39	5	144	0.985	58	66
		80	5	1441	4997	0	352	1649	372	59	1015	0.983	327	1756
	1.67	20	5	14	6	0	18	0	0	0	8	0.999	1	1
		40	5	69	32	0	17	0	0	0	56	0.996	15	3
		60	5	174	269	0	74	1	1	0	190	0.988	60	43
		80	5	253	249	0	14	7	6	0	357	0.991	75	106
		100	4	279	278	0	14	16	9	1	316	0.993	57	194
0.5	1	20	5	16	23	0	15	13	5	1	9	0.996	3	1
		40	5	53	150	0	21	31	15	1	59	0.989	14	4
		60	5	350	1120	0	133	247	99	9	267	0.987	148	184
		80	5	654	1120	0	76	439	89	10	708	0.987	183	397
	1.33	20	5	16	23	0	15	13	5	1	9	0.996	3	1
		40	5	53	150	0	21	31	15	1	59	0.989	14	4
		60	5	350	1120	0	133	247	99	9	267	0.987	148	184
		80	5	654	1120	0	76	439	89	10	708	0.987	183	404
	1.67	20	5	7	7	0	6	1	1	0	5	0.999	1	1
		40	5	72	80	0	13	1	1	0	68	0.988	18	4
		60	5	101	118	0	10	5	4	2	121	0.991	31	18
		80	5	822	1975	0	135	32	24	5	862	0.987	360	683
		100	3	221	85	0	2	8	8	3	261	0.996	14	38

Table 5 Summary of computation results for TSPLIB instances for $\alpha = 0.35$, $\beta = 1.33$ and $\gamma = \infty$

TSPLIB Instance	C0	C4	C8	C9	C14	C15	C16	COMB	LB/z*	NODES	SEC
burma14	4	5	5	0	4	0	0	0	1.000	1	1
ulysse16	23	18	54	51	14	4	2	9	0.986	3	1
gr17	25	42	29	45	22	8	5	4	0.997	13	1
gr21	22	15	0	47	10	5	1	0	1.000	1	1
ulysses22	23	24	0	39	2	2	0	21	1.000	3	1
gr24	29	129	0	166	33	5	0	13	0.981	19	2
fri26	41	43	0	95	19	6	2	43	0.991	9	1
bays29	90	372	0	194	16	4	0	43	0.983	37	6
dantzig42	159	552	0	167	141	33	1	71	0.974	189	64
swiss42	89	102	0	23	11	3	0	66	0.998	3	2
att48	8819	53467	0	18175	8848	4167	441	1172	0.964	4471	7602
gr48	267	1115	0	245	291	79	11	240	0.976	211	183
hk48	41	123	0	68	42	29	7	37	0.997	3	3
eil51	3808	12973	0	1949	5122	1124	190	2740	0.977	1065	1108
berlin52	64	231	0	57	33	16	1	88	1.000	1	5
brazil58	18	204	0	22	53	15	1	0	0.997	15	10
eil76	816	2504	0	203	1171	206	54	580	0.991	231	647
rat99	4708	11435	0	1156	4292	735	66	5345	0.989	1379	7350
kroC100	1721	2179	0	166	881	168	2	2335	0.976	699	2376
kroD100	621	2798	0	182	690	176	25	570	0.984	475	2171
kroE100	647	621	0	28	433	77	6	859	0.986	159	359

3.3 Comparisons with the TSP and the unit demand VRP

Finally, since the TSP and the unit demand VRP are special cases of the BWTSP, we have conducted some tests to see how the BWTSP compares with these two problems. Our results are reported in Table 6 for the TSP and in Table 7 for the VRP. As expected, they indicate that the BWTSP is much more difficult than the corresponding TSP over the same number of vertices. This is due to the introduction of cardinality and length restrictions which requires the generation of significantly more constraints throughout the branch-and-cut process. This is particularly true of comb inequalities which are far more numerous in the BWTSP than in the TSP. Table 6 shows that our BWTSP algorithm was able to solve to optimality and within reasonable computing times most unit demand VRPs involving up to 60 to 100 vertices, depending on α. These results seem to compare well with those obtained by Araque et al. (1994) who solved some instances of the same problem with an unbounded number of vehicles for $21 \leq n \leq 60$.

Table 6 Application of the BWTSP algorithm to the TSP

n	SUCC	CO	2M	LB/z^*	NODES	SEC
20	5	13	4	0.997	2	0
40	5	89	51	0.996	17	1
60	5	262	162	0.991	277	63
80	5	235	145	0.994	158	49
100	4	209	116	0.995	168	86

Table 7 Application of the BWTSP algorithm to the unit demand VRP ($\beta = 1, \gamma = \infty$)

α	n	SUCC	CO	C4	C8	C9	C14	C15	C16	2M	NODES	LB/z^*	SEC
0.2	20	5	22	70	145	33	25	13	4	4	6	0.998	3
	40	5	81	1300	217	599	293	100	20	40	18	0.994	158
	60	4	170	1631	0	630	361	113	21	130	25	0.995	322
0.35	20	5	16	44	0	77	13	6	1	13	1	1	1
	40	5	63	278	0	85	55	18	2	79	14	0.992	35
	60	5	191	1493	0	125	297	70	6	154	105	0.987	613
	80	4	603	3143	0	157	955	231	22	708	199	0.987	3023
0.5	20	5	11	27	0	23	6	3	1	16	1	1	0
	40	5	46	168	0	26	53	22	2	53	13	0.996	5
	60	5	105	217	0	24	78	29	3	147	31	0.989	29
	80	5	349	1464	0	95	542	178	13	399	109	0.987	401
	100	4	917	3301	0	171	1563	451	36	1101	207	0.990	1580

4. Conclusions

In this article, we have introduced the *Black and White Traveling Salesman Problem*, an extension of the classical TSP with applications in the fields of short-haul airline scheduling and telecommunications. The undirected version of the problem was formulated as an integer linear program and several classes of valid inequalities were derived. An exact branch-and-cut algorithm was then developed. Extensive computational experiments have shown that medium size instances (involving up to 100 vertices) can be solved optimally. The algorithm can also solve exactly unit demand VRPs of similar sizes.

Appendix : An exact separation procedure for constraints (8)

We have developed the following exact separation for constraints (8). Given a fractional solution $\left(x^*_{ij}\right)$, determine the subset $S \subseteq W$ corresponding to the most violated inequality (8) (if any) by means of the following procedure:

Step 1: Construct a graph $G' = \left(W \cup \{v_s, v_t\}, \tilde{E}\right)$, where edge $(v_i, v_j) \in \tilde{E}$ has a capacity u_{ij} equal to

$$\begin{cases} (1-2/Q)x^*_{ij} & \text{if} \quad v_i, v_j \in W \\ \max\left\{0, \sum\limits_{v_p \in B} x^*_{pj} - 2/Q\right\} & \text{if} \quad v_i = v_s \text{ and } v_j \in W \\ \max\left\{0, 2/Q - \sum\limits_{v_p \in B} x^*_{pj}\right\} & \text{if} \quad v_i = v_t \text{ and } v_j \in W. \end{cases}$$

Step 2: Determine the capacity U of the minimum cut $(\{v_t\} \cup W_1, \{v_s\} \cup W_2)$ separating v_t and v_s, where $\{W_1, W_2\}$ is a partition of W. If $U - \sum\limits_{v_j \in W} u_{tj} < 0$, then constraint (8) corresponding to $S = W_1$ is the most violated inequality (8). Otherwise no violated constraint (8) exists.

The validity of this procedure is established as follows. The capacity of the cut edge set $(\{v_t\} \cup W_1, \{v_s\} \cup W_2)$ minus the constant $\sum\limits_{v_j \in W} u_{tj}$ can be expressed as $\sum\limits_{\substack{v_i \in W_1 \\ v_j \in W_2}} u_{ij} + \sum\limits_{v_j \in W_1} u_{sj} + \sum\limits_{v_j \in W_2} u_{tj} -$

$$\sum_{v_j \in W} u_{tj} = \sum_{\substack{v_i \in W_1 \\ v_j \in W_2}} (1-2/Q)x^*_{ij} + \sum_{v_j \in W_1} \left(\max\left\{0, \sum_{v_p \in B} x^*_{pj} - 2/Q\right\} - \max\left\{0, 2/Q - \sum_{v_p \in B} x^*_{pj}\right\}\right)$$

$= (1-2/Q)x(\delta_W(W_1))+x(\delta_B(W_1)) - 2|W_1|/Q$ and is therefore proportional to the slack of inequality (8) for $S = W_1$. If $S = \varnothing$, we apply the same procedure $|W|$ times, forcing each time a vertex $v_i \in W$ to belong to S by making $u_{is} = \infty$, and using all the resulting sets to check inequality (8).□

Acknowledgements

This research was partially funded by the Canadian Natural Sciences and Engineering Research Council under grants OGP0039682 and OGP0184123, by the Ministero dell'Università e della Ricerca Scientifica e Tecnologica, Italy (progetto MOST), and by the Région Nord-Pas de Calais, France. This support is gratefully acknowledged. Thanks are due to the Area Editor, to the Associate Editor and to the referees for their valuable comments.

References

Araque, J.R., G. Kudva, T.L. Morin, J.F. Pekny. 1994. A Branch-and-Cut Algorithm for Vehicle Routing Problems. *Annals of Operations Research* 50, 37–59.

Augerat, P., J.M. Belenguer, E. Benavent, A. Corberán, D. Naddef. 1998. Separating Capacity Constraints in the CVRP Using Tabu Search. *European Journal of Operational Research* 106, 546–557.

Bourgeois, M., G. Laporte, F. Semet. 2003. Heuristics for the Black and White Traveling Salesman Problem. *Computers & Operations Research* 30, 75–85.

Cosares, S., D.N. Deutsch, I. Saniee, O.J. Wasem. 1995. SONET Toolkit: A Decision Support System for Designing Robust and Cost-Effective Fiber-Optic Networks. *Interfaces* 25(1), 20–40.

Caprara, A., M. Fischetti. 1997. Branch-and-Cut Algorithms. In M. Dell'Amico, F. Maffioli, S. Martello, eds., *Annotated Bibliographies in Combinatorial Optimization*, Wiley, Chichester, 45–63.

Dantzig, G.B., D.R. Fulkerson, S.M. Johnson. 1954. Solution of a Large-Scale Traveling-Salesman Problem. *Operations Research* 2, 393–410.

Fischetti, M., J.J. Salazar, P. Toth. 1997. A Branch-and-Cut Algorithm for the Symmetric Generalized Traveling Salesman Problem. *Operations Research* 45, 378–394.

Fischetti, M., P. Toth. 1997. A Polyhedral Approach to the Asymmetric Traveling Salesman Problem. *Management Science* 43, 1520–1536.

Gendreau, M., A. Hertz, G. Laporte. 1992. New Insertion and Postoptimization Procedures for the Traveling Salesman Problem. *Operations Research* 40, 1086–1094.

Gendreau, M., A. Hertz, G. Laporte. 1994. A Tabu Search Algorithm for the Vehicle Routing Problem. *Management Science* 40, 1276–1290.

Glover, F., M. Laguna. 1997. *Tabu Search*. Kluwer, Boston.

Grötschel, M., M.W. Padberg. 1985. Polyhedral Theory. *The Traveling Salesman Problem*. E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, D.B. Shmoys, eds. Wiley, Chichester.

Held, M., R.M. Karp. 1971. The Traveling Salesman Problem and Minimum Spanning Trees: Part II. *Mathematical Programming* 1, 6–25.

Laporte, G., Y. Nobert, M. Desrochers. 1985. Optimal Routing under Capacity and Distance Restrictions. *Operations Research* 33, 1050–1073.

Lenstra, J.K., A.H.G. Rinnooy Kan. 1975. Some Simple Applications of the Traveling Salesman Problem. *Operational Research Quarterly* 26, 717–733.

Mak, V., N. Boland. 2000. Heuristic Approaches to the Asymmetric Travelling Salesman Problem with Replenishment Arcs. *International Transactions in Operational Research* 7, 431–447.

Naddef, D., G. Rinaldi. 1993. The Graphical Relaxation: A New Framework for the Symmetric Traveling Salesman Polytope. *Mathematical Programming* 58, 53–87.

Padberg, M.W., M. Grötschel. 1985. Polyhedral Computations. In E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, D.B. Shmoys, eds., *The Traveling Salesman Problem. A Guided Tour of Combinatorial Optimization*, Wiley, Chichester, 307–360.

Padberg, M.W., G. Rinaldi. 1990. Facet Identification for the Symmetric Traveling Salesman Polytope. *Mathematical Programming* 47, 219–257.

Reinelt, G. 1991. TSPLIB — A Traveling Salesman Problem Library. ORSA *Journal on Computing* 3, 376–384.

Taillard, É.D. 1991. Robust Taboo Search for the Quadratic Assignment Problem. *Parallel Computing* 17, 443–455.

Talluri, K.T. 1998. The Four-Day Aircraft Maintenance Routing Problem. *Transportation Science* 32, 43–53.

Volgenant, T., R. Jonker. 1982. A Branch and Bound Algorithm for the Symmetric Traveling Salesman Problem Based on the 1-Tree Relaxation. *European Journal of Operational Research* 9, 83–89.

Wasem, O.J. 1991. An Algorithm for Designing Rings in Survivable Fiber Networks. *IEEE Transactions on Reliability* 40, 428–432.